

Lab 1

Jordan Small

October 1, 2023

CDA3203 Computer Logic Design




Fall 2023

Dr. Maria Petrie



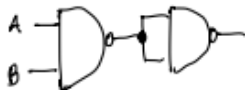
Florida Atlantic University

Part 1: Design 9 circuits by completing below: Draw the symbol for the gate, its Truth Table, its Simplest Sum of Products Expression, draw its NOT-AND-OR Equivalent Circuit, its all-NAND Equivalent Circuit.



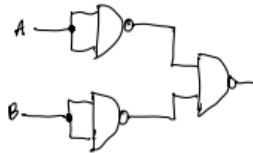
1.1 NOT gate.

Draw NOT gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit						
	<table border="1"><thead><tr><th>A</th><th>NOT(A)</th></tr></thead><tbody><tr><td>1</td><td>0</td></tr><tr><td>0</td><td>1</td></tr></tbody></table> <p>$Y1 = A'$</p>	A	NOT(A)	1	0	0	1		
A	NOT(A)								
1	0								
0	1								

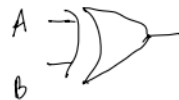
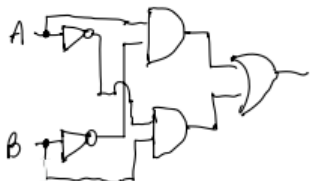
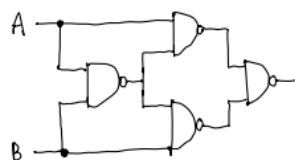
1.2- AND gate

Draw AND gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit															
	<table border="1"><thead><tr><th>A</th><th>B</th><th>AND(A,B)</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></tbody></table> <p>Y2 = AB</p>	A	B	AND(A,B)	0	0	0	0	1	0	1	0	0	1	1	1		
A	B	AND(A,B)																
0	0	0																
0	1	0																
1	0	0																
1	1	1																

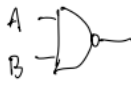
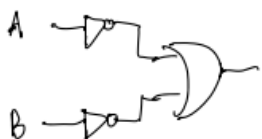

1.3- OR gate

Draw OR gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit															
	<table border="1" data-bbox="444 449 704 638"><thead><tr><th>A</th><th>B</th><th>OR(A,B)</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></tbody></table> <p data-bbox="487 680 605 703">$Y3 = A + B$</p>	A	B	OR(A,B)	0	0	0	0	1	1	1	0	1	1	1	1		
A	B	OR(A,B)																
0	0	0																
0	1	1																
1	0	1																
1	1	1																


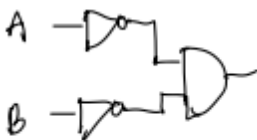
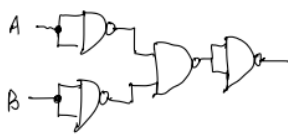
1.4- XOR gate

Draw XOR gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit															
	<table border="1" data-bbox="433 1047 709 1236"><thead><tr><th>A</th><th>B</th><th>XOR(A,B)</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></tbody></table> <p data-bbox="490 1268 657 1293">$Y4 = AB' + A'B$</p>	A	B	XOR(A,B)	0	0	0	0	1	1	1	0	1	1	1	0		
A	B	XOR(A,B)																
0	0	0																
0	1	1																
1	0	1																
1	1	0																

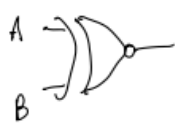
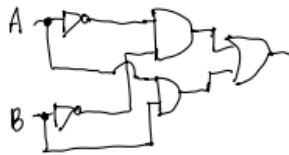
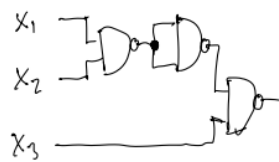
1.5- NAND gate

Draw NAND gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit															
	<table><tr><th>A</th><th>B</th><th>NAND(A,B)</th></tr><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></table> <p>Y5 = A' + B'</p>	A	B	NAND(A,B)	0	0	1	0	1	1	1	0	1	1	1	0		
A	B	NAND(A,B)																
0	0	1																
0	1	1																
1	0	1																
1	1	0																

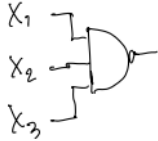
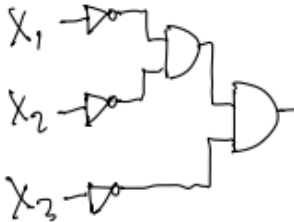
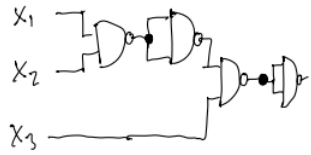
1.6- NOR gate

Draw NOR gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit															
	<table border="1" data-bbox="435 1010 711 1197"><thead><tr><th>A</th><th>B</th><th>NOR(A,B)</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>0</td></tr></tbody></table> <p data-bbox="500 1239 605 1264">$Y6 = A'B'$</p>	A	B	NOR(A,B)	0	0	1	0	1	0	1	0	0	1	1	0		
A	B	NOR(A,B)																
0	0	1																
0	1	0																
1	0	0																
1	1	0																

1.7- XNOR gate

Draw XNOR gate	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit															
	<table border="1" data-bbox="433 445 721 630"><thead><tr><th>A</th><th>B</th><th>XNOR(A,B)</th></tr></thead><tbody><tr><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></tbody></table> <p data-bbox="487 644 654 672">$Y7 = AB + A'B'$</p>	A	B	XNOR(A,B)	0	0	1	0	1	0	1	0	0	1	1	1		
A	B	XNOR(A,B)																
0	0	1																
0	1	0																
1	0	0																
1	1	1																

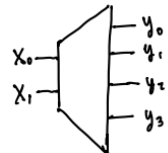
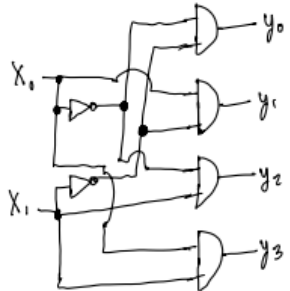
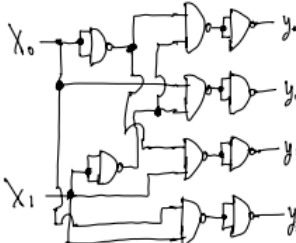
1.8 3-input NAND gate

Draw 3-input NAND	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit																																				
	<table border="1" data-bbox="441 1029 714 1360"> <thead> <tr> <th>X_2</th><th>X_1</th><th>X_0</th><th>NAND3</th></tr> </thead> <tbody> <tr> <td>0</td><td>0</td><td>0</td><td>1</td></tr> <tr> <td>0</td><td>0</td><td>1</td><td>1</td></tr> <tr> <td>0</td><td>1</td><td>0</td><td>1</td></tr> <tr> <td>0</td><td>1</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>0</td><td>1</td></tr> <tr> <td>1</td><td>0</td><td>1</td><td>1</td></tr> <tr> <td>1</td><td>1</td><td>0</td><td>1</td></tr> <tr> <td>1</td><td>1</td><td>1</td><td>0</td></tr> </tbody> </table> <p data-bbox="479 1409 672 1436">$Y8 = A' + B' + C'$</p>	X_2	X_1	X_0	NAND3	0	0	0	1	0	0	1	1	0	1	0	1	0	1	1	1	1	0	0	1	1	0	1	1	1	1	0	1	1	1	1	0		
X_2	X_1	X_0	NAND3																																				
0	0	0	1																																				
0	0	1	1																																				
0	1	0	1																																				
0	1	1	1																																				
1	0	0	1																																				
1	0	1	1																																				
1	1	0	1																																				
1	1	1	0																																				

1.9- 2 to 1 Encoder or Multiplexer (Mux)

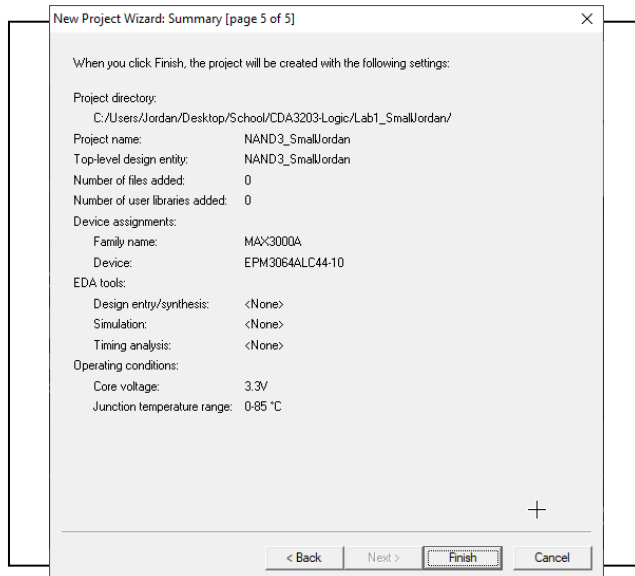
Draw 2to1 Mux	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit																																				
	<table border="1"> <thead> <tr> <th>s</th><th>X₁</th><th>X₂</th><th>Mux</th></tr> </thead> <tbody> <tr><td>0</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>0</td><td>0</td><td>1</td><td>0</td></tr> <tr><td>0</td><td>1</td><td>0</td><td>1</td></tr> <tr><td>0</td><td>1</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>0</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>0</td><td>1</td><td>1</td></tr> <tr><td>1</td><td>1</td><td>0</td><td>0</td></tr> <tr><td>1</td><td>1</td><td>1</td><td>1</td></tr> </tbody> </table> $Y_8 = S'X_1 + SX_2$	s	X ₁	X ₂	Mux	0	0	0	0	0	0	1	0	0	1	0	1	0	1	1	1	1	0	0	0	1	0	1	1	1	1	0	0	1	1	1	1		
s	X ₁	X ₂	Mux																																				
0	0	0	0																																				
0	0	1	0																																				
0	1	0	1																																				
0	1	1	1																																				
1	0	0	0																																				
1	0	1	1																																				
1	1	0	0																																				
1	1	1	1																																				

1.10 - 2 to 4 decoder or Demultiplexer (DMux)

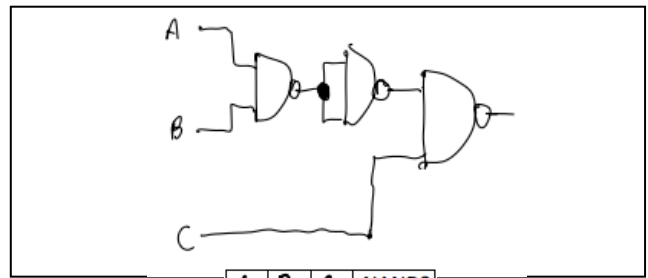
Draw 2to4 DMux	Truth Table and Simplest Sum of Products Equation	NOT-AND-OR Equivalent Circuit	all-NAND Equivalent Circuit																														
	<table border="1" data-bbox="412 1287 782 1474"><tr><th>X_1</th><th>X_0</th><th>Y_3</th><th>Y_2</th><th>Y_1</th><th>Y_0</th></tr><tr><td>0</td><td>0</td><td>0</td><td>0</td><td>0</td><td>1</td></tr><tr><td>0</td><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td><td>0</td><td>0</td><td>0</td></tr></table> <p>$Y_3 = X_1X_0$</p> <p>$Y_2 = X_1X_0'$</p> <p>$Y_1 = X_1'X_0$</p> <p>$Y_0 = X_1'X_0'$</p>	X_1	X_0	Y_3	Y_2	Y_1	Y_0	0	0	0	0	0	1	0	1	0	0	1	0	1	0	0	1	0	0	1	1	1	0	0	0		
X_1	X_0	Y_3	Y_2	Y_1	Y_0																												
0	0	0	0	0	1																												
0	1	0	0	1	0																												
1	0	0	1	0	0																												
1	1	1	0	0	0																												

2.1 Create a new project in Altera Quartus using VHDL of a **3-input NAND** circuit call it NAND3_YourName, **using all-NAND gates**

Project Wizard



3-input NAND Diagram

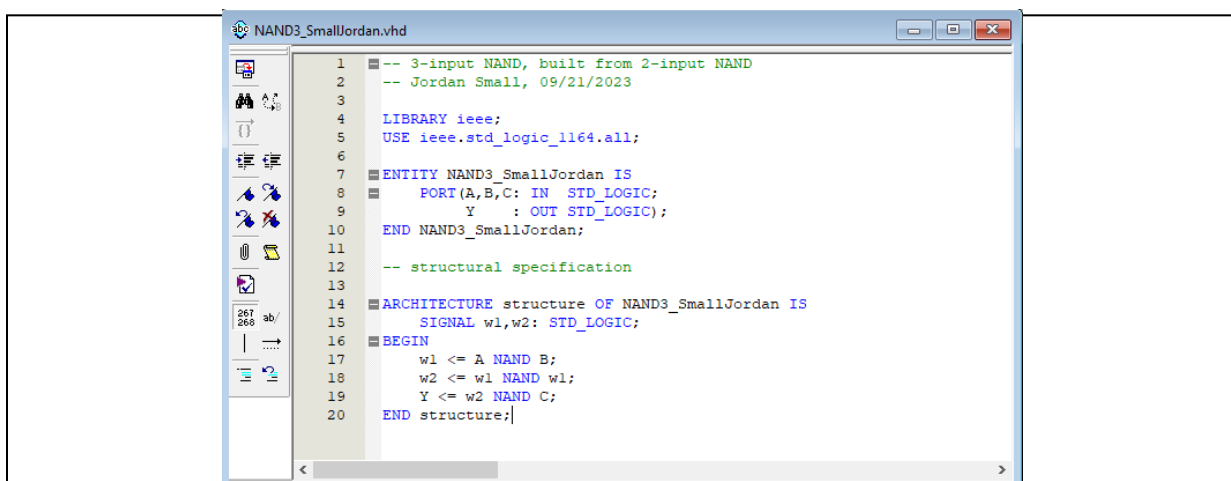


Truth Table

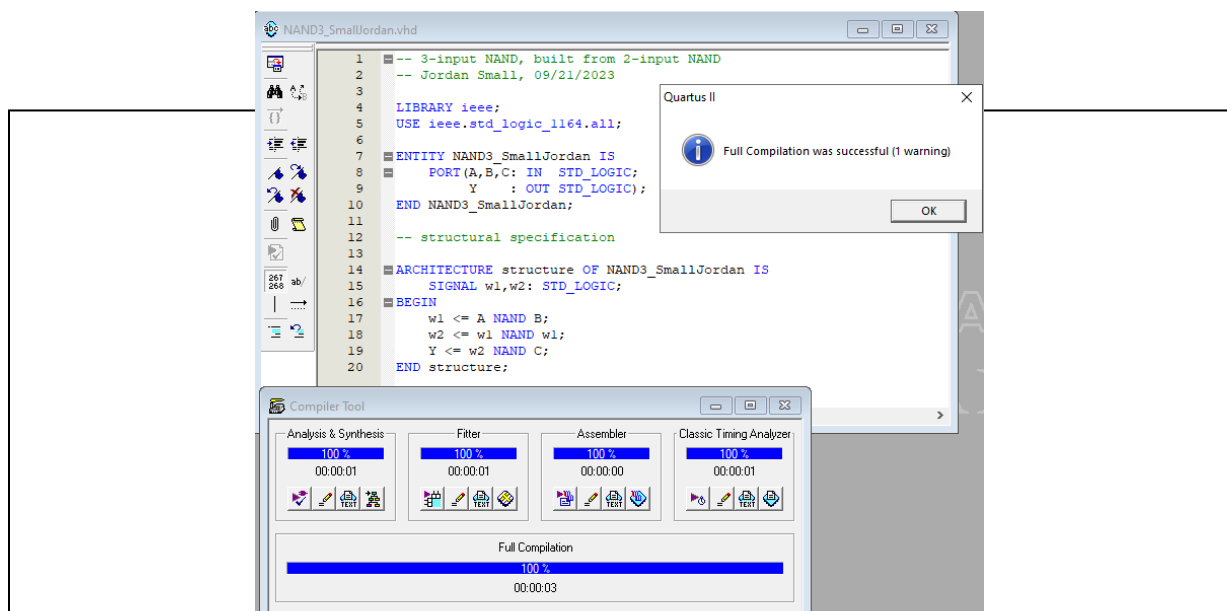
A	B	C	NAND3
0	0	0	1
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	1
1	1	1	0

$$Y = A + B + C$$

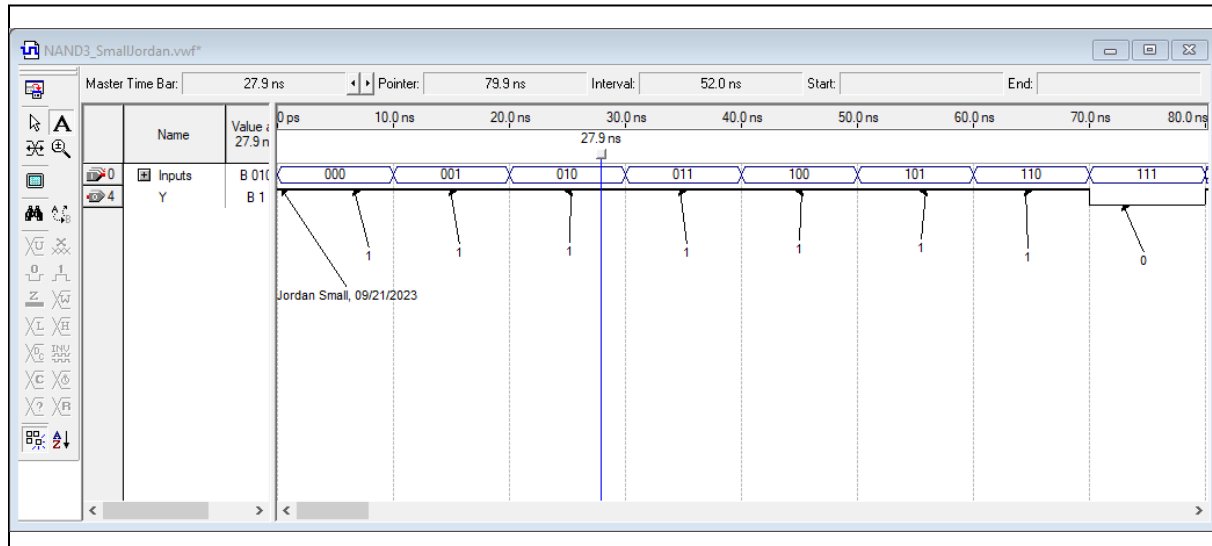
VHDL code



Successful Compilation



Timing Diagram



2.2 Create a new project in Altera Quartus using VHDL of a **4-input NAND** circuit, call it NAND4_YourName, **using all-NAND gates**

Project Wizard

New Project Wizard: Summary [page 5 of 5]

When you click Finish, the project will be created with the following settings:

Project directory:
C:/Users/Jordan/Desktop/School/CDA3203-Logic/Lab1_SmallJordan/

Project name:
NAND4_SmallJordan

Top-level design entity:
NAND4_SmallJordan

Number of files added:
0

Number of user libraries added:
0

Device assignments:
Family name:
MAX3000A
Device:
EPM3064ALC44-10

EDA tools:
Design entry/synthesis:
<None>
Simulation:
<None>
Timing analysis:
<None>

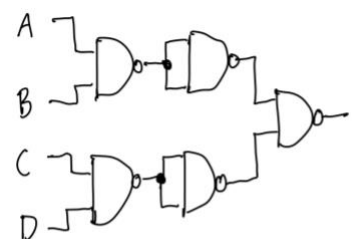
Operating conditions:
Core voltage:
3.3V
Junction temperature range:
0-85 °C

< Back Next > Finish Cancel

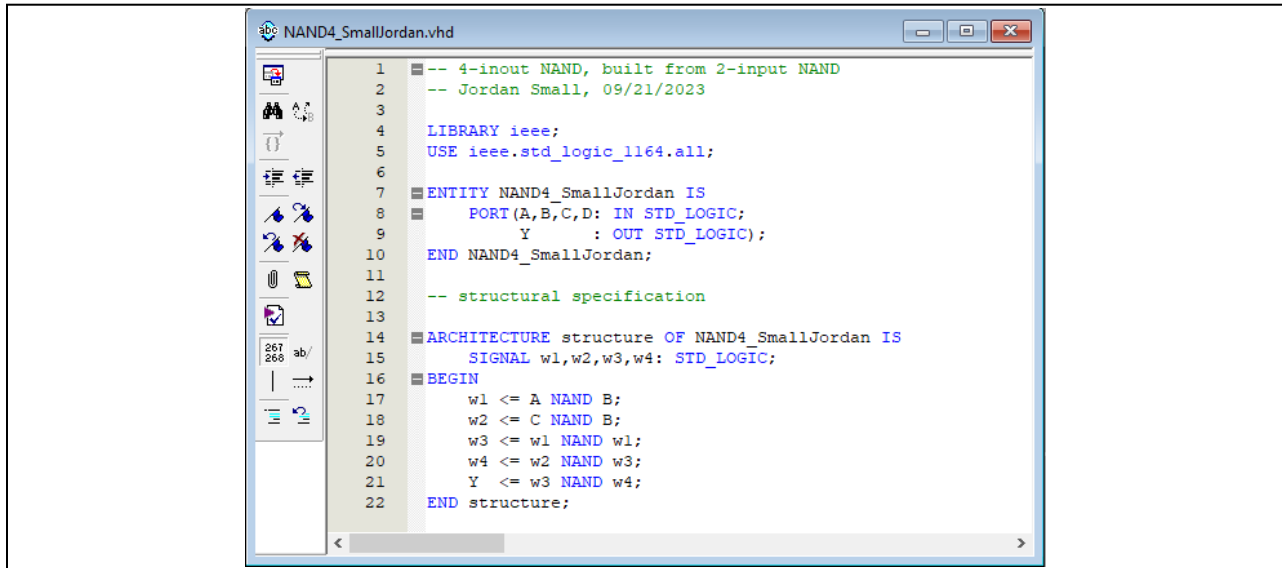
Truth Table

A	B	C	D	Y
0	0	0	0	1
0	0	0	1	1
0	0	1	0	1
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

4-input NAND Diagram

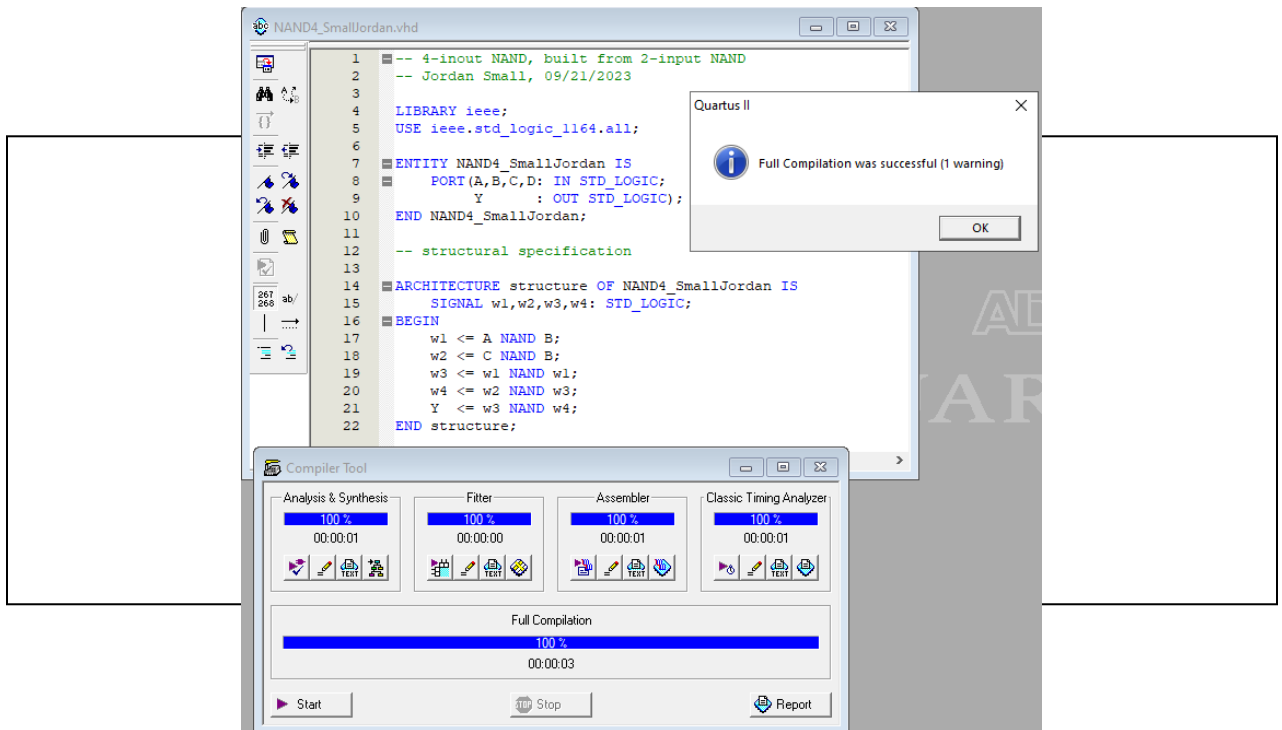


VHDL code

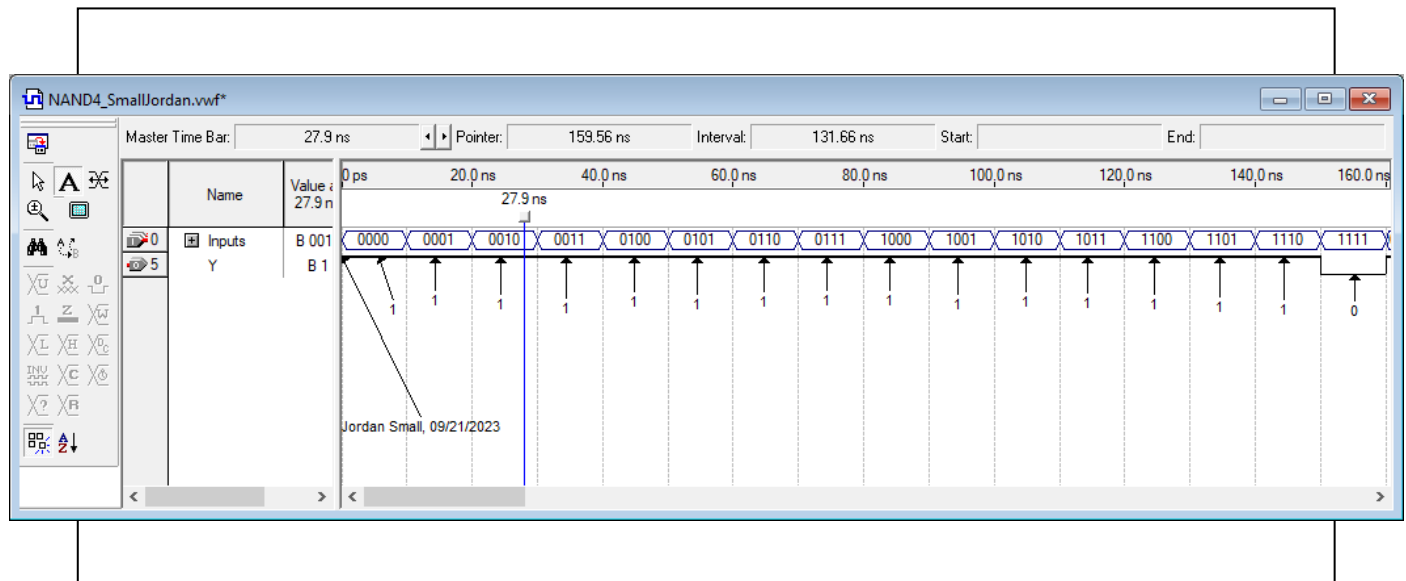


```
1  -- 4-inout NAND, built from 2-input NAND
2  -- Jordan Small, 09/21/2023
3
4  LIBRARY ieee;
5  USE ieee.std_logic_1164.all;
6
7  ENTITY NAND4_SmallJordan IS
8      PORT (A,B,C,D: IN STD_LOGIC;
9            Y      : OUT STD_LOGIC);
10 END NAND4_SmallJordan;
11
12 -- structural specification
13
14 ARCHITECTURE structure OF NAND4_SmallJordan IS
15     SIGNAL w1,w2,w3,w4: STD_LOGIC;
16 BEGIN
17     w1 <= A NAND B;
18     w2 <= C NAND B;
19     w3 <= w1 NAND w1;
20     w4 <= w2 NAND w3;
21     Y  <= w3 NAND w4;
22 END structure;
```

Successful Compilation

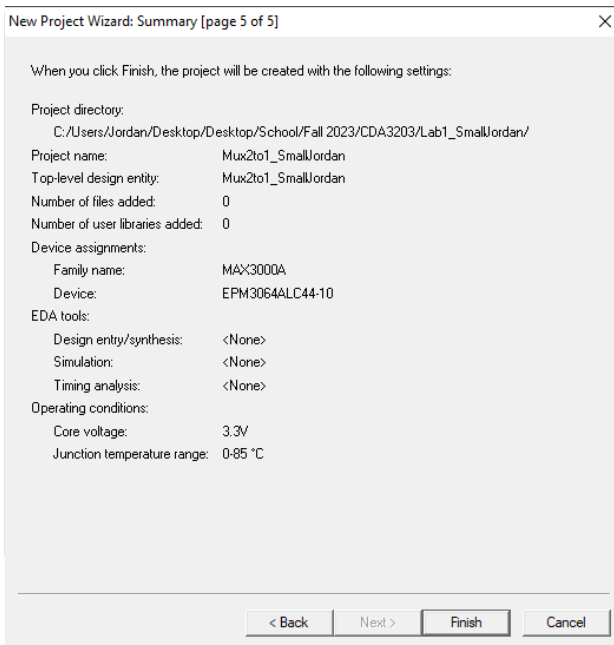


Timing Diagram



2.3 Create a new project in Altera Quartus using VHDL of a **2 to 1 Multiplexer/Encoder (Mux)** circuit, call it Mux2to1_YourName, **using all-NAND gates**

Project Wizard

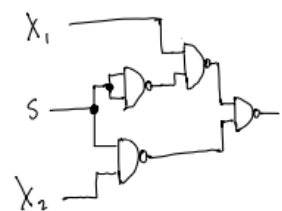


Truth Table

S	X ₁	X ₂	Mux
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	1

$$Y = S'X_1 + SX_2$$

Mux2to1 Diagram



VHDL code

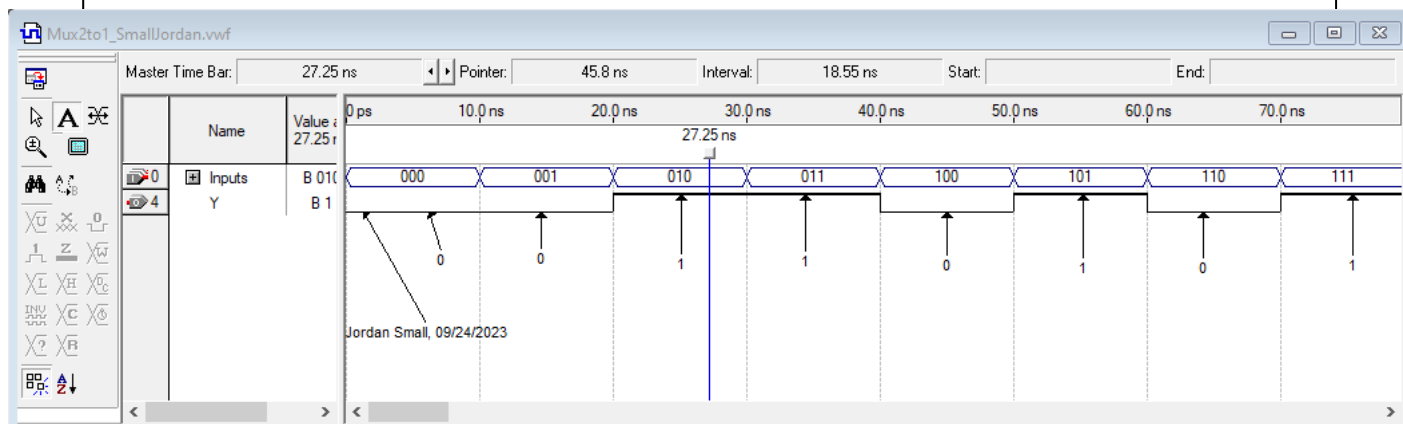
```
Mux2to1_SmallJordan.vhd

1  -- Mux2to1, built from 2-input NAND
2  -- Jordan Small, 09/24/2023
3
4  LIBRARY ieee;
5  USE ieee.std_logic_1164.all;
6
7  ENTITY Mux2to1_SmallJordan IS
8  PORT(X1,X2,S: IN  STD_LOGIC;
9        Y      : OUT STD_LOGIC);
10 END Mux2to1_SmallJordan;
11
12 --structural specification
13
14 ARCHITECTURE structure OF Mux2to1_SmallJordan IS
15     SIGNAL w1,w2,w3: STD_LOGIC;
16 BEGIN
17     w1 <= S NAND S;
18     w2 <= X1 NAND w1;
19     w3 <= S NAND X2;
20     Y  <= w2 NAND w3;
21 END structure;
```

Successful Compilation

The image shows the Quartus II Compiler Tool window. A message box titled "Quartus II" displays the text "Full Compilation was successful (1 warning)" with an "OK" button. Below this, the Compiler Tool window shows progress bars for Analysis & Synthesis (100%), Filter (100%), Assembler (100%), and Classic Timing Analyzer (100%). The Full Compilation progress bar is also at 100%. The Master Time Bar shows a total time of 00:00:04. Buttons for "Stop" and "Report" are visible at the bottom.

Timing Diagram



2.4. Create a new project in Altera Quartus using VHDL of a **4-to-1 Mux** circuit, call it Mux4to1_YourName, **using only Mux2to1 components**.

Project Wizard

New Project Wizard: Summary [page 5 of 5]

When you click Finish, the project will be created with the following settings:

Project directory:
C:/Users/Jordan/Desktop/School/Fall 2023/CDA3203/Lab1_SmallJordan/

Project name: Mux4to1_SmallJordan
Top-level design entity: Mux4to1_SmallJordan
Number of files added: 1
Number of user libraries added: 0

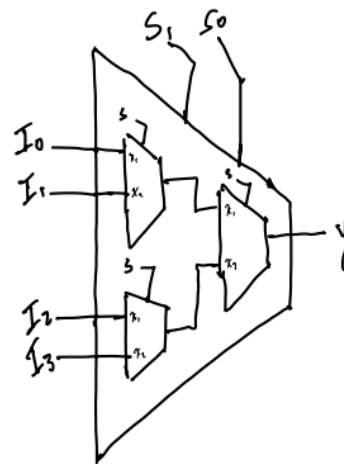
Device assignments:
Family name: MAX3000A
Device: EPM3064ALC44-10

EDA tools:
Design entry/synthesis: <None>
Simulation: <None>
Timing analysis: <None>

Operating conditions:
Core voltage: 3.3V
Junction temperature range: 0-85 °C

< Back Next > Finish Cancel

Mux4to1 Diagram



Truth Table

S_1	S_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

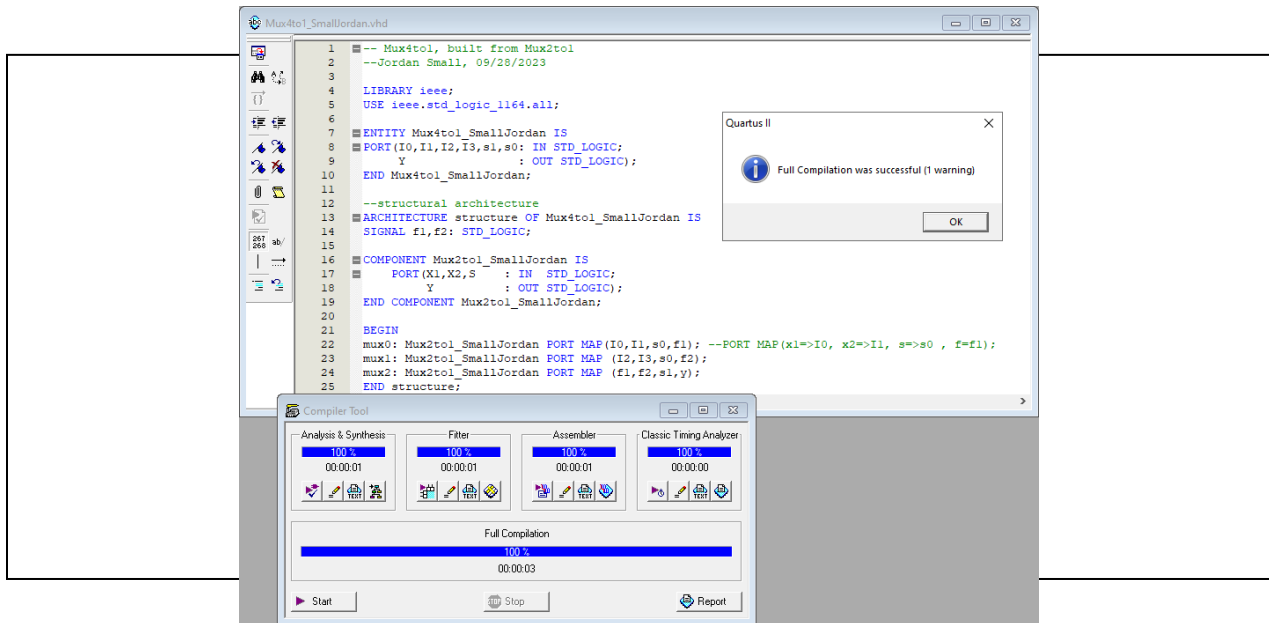
VHDL code

```

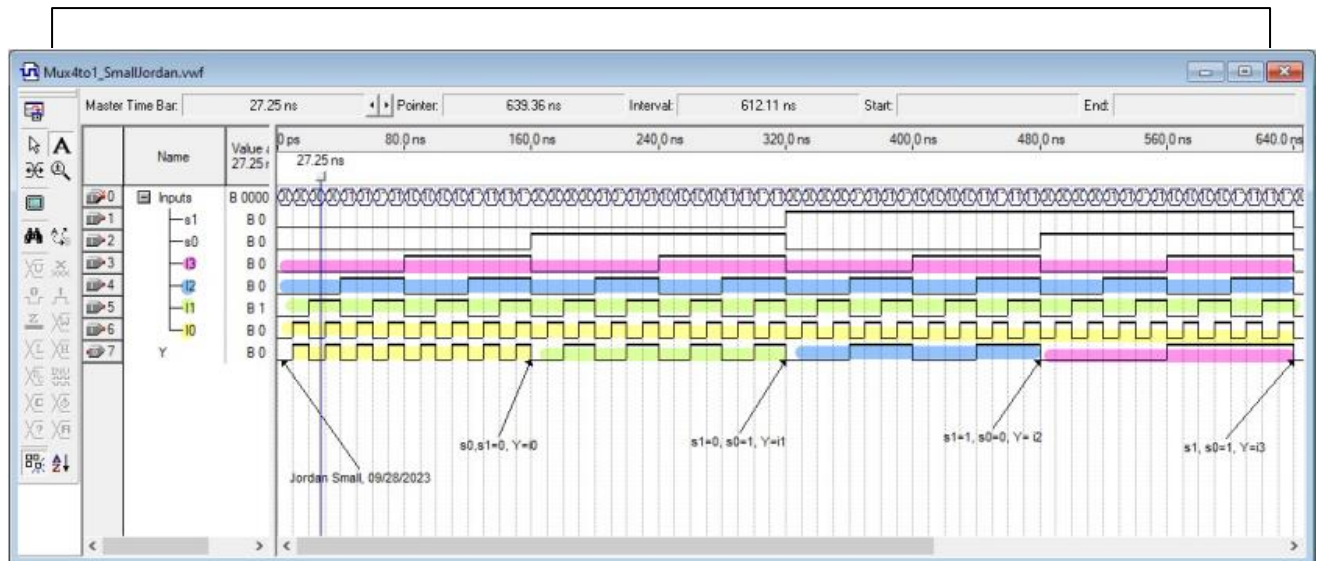
1  -- Mux4to1, built from Mux2to1
2  --Jordan Small, 09/28/2023
3
4  LIBRARY ieee;
5  USE ieee.std_logic_1164.all;
6
7  ENTITY Mux4to1_SmallJordan IS
8  PORT(I0,I1,I2,I3,s1,s0: IN STD_LOGIC;
9        Y                : OUT STD_LOGIC);
10 END Mux4to1_SmallJordan;
11
12 --structural architecture
13 ARCHITECTURE structure OF Mux4to1_SmallJordan IS
14 SIGNAL f1,f2: STD_LOGIC;
15
16 COMPONENT Mux2to1_SmallJordan IS
17 PORT(X1,X2,S      : IN STD_LOGIC;
18       Y           : OUT STD_LOGIC);
19 END COMPONENT Mux2to1_SmallJordan;
20
21 BEGIN
22 mux0: Mux2to1_SmallJordan PORT MAP(I0,I1,s0,f1); --PORT MAP(x1=>I0, x2=>I1, s=>s0 , f=f1);
23 mux1: Mux2to1_SmallJordan PORT MAP (I2,I3,s0,f2);
24 mux2: Mux2to1_SmallJordan PORT MAP (f1,f2,s1,y);
25 END structure;

```

Successful Compilation



Timing Diagram



2.5 Create a new project in Altera Quartus using VHDL of a **8-to-1 Mux**, call it Mux8to1_YourName, using **only Mux4to1 and/or Mux2to1 components**

Project Wizard

New Project Wizard: Summary [page 5 of 5]

When you click Finish, the project will be created with the following settings:

Project directory:
C:/Users/Jordan/Desktop/School/Fall 2023/CDA3203/Lab1_SmallJordan/

Project name: Mux8to1_SmallJordan

Top-level design entity: Mux8to1_SmallJordan

Number of files added: 2

Number of user libraries added: 0

Device assignments:

Family name: MAX3000A

Device: EPM3064ALC44-10

EDA tools:

Design entry/synthesis: <None>

Simulation: <None>

Timing analysis: <None>

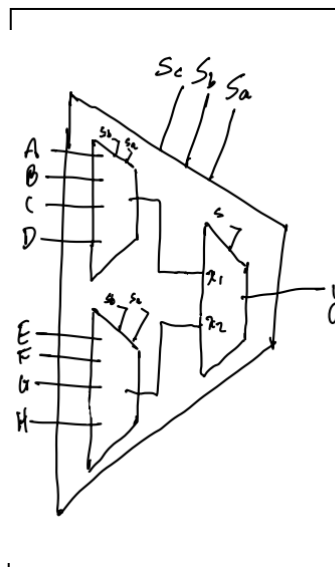
Operating conditions:

Core voltage: 3.3V

Junction temperature range: 0-85 °C

< Back Next > Finish Cancel

Mux8to1 Diagram



Truth Table

Sc	Sb	Sa	y
0	0	0	A
0	0	1	B
0	1	0	C
0	1	1	D
1	0	0	E
1	0	1	F
1	1	0	G
1	1	1	H

VHDL code

```

1  -- Mux8to1, built from Mux2to1 and Mux4to1
2  --Jordan Small, 09/30/2023
3
4  LIBRARY ieee;
5  USE ieee.std_logic_1164.all;
6
7  ENTITY Mux8to1_SmallJordan IS
8  PORT (A,B,C,D,E,F,G,H,Sc,Sb,Sa: IN STD_LOGIC;
9        Y: OUT STD_LOGIC);
10 END Mux8to1_SmallJordan;
11
12 --structural architecture
13 ARCHITECTURE structure OF Mux8to1_SmallJordan IS
14 SIGNAL f1,f2: STD_LOGIC;
15
16 COMPONENT Mux2to1_SmallJordan IS
17 PORT (X1,X2,S: IN STD_LOGIC;
18       Y: OUT STD_LOGIC);
19 END COMPONENT Mux2to1_SmallJordan;
20
21 COMPONENT Mux4to1_SmallJordan IS
22 PORT (I0,I1,I2,I3,s1,s0: IN STD_LOGIC;
23       Y: OUT STD_LOGIC);
24 END COMPONENT Mux4to1_SmallJordan;
25
26 BEGIN
27 muxA: Mux4to1_SmallJordan PORT MAP (A,B,C,D,Sa,Sb,f1);
28 muxB: Mux4to1_SmallJordan PORT MAP (E,F,G,H,Sa,Sb,f2);
29 muxC: Mux2to1_SmallJordan PORT MAP (f1,f2,Sc,y);
30 END structure;

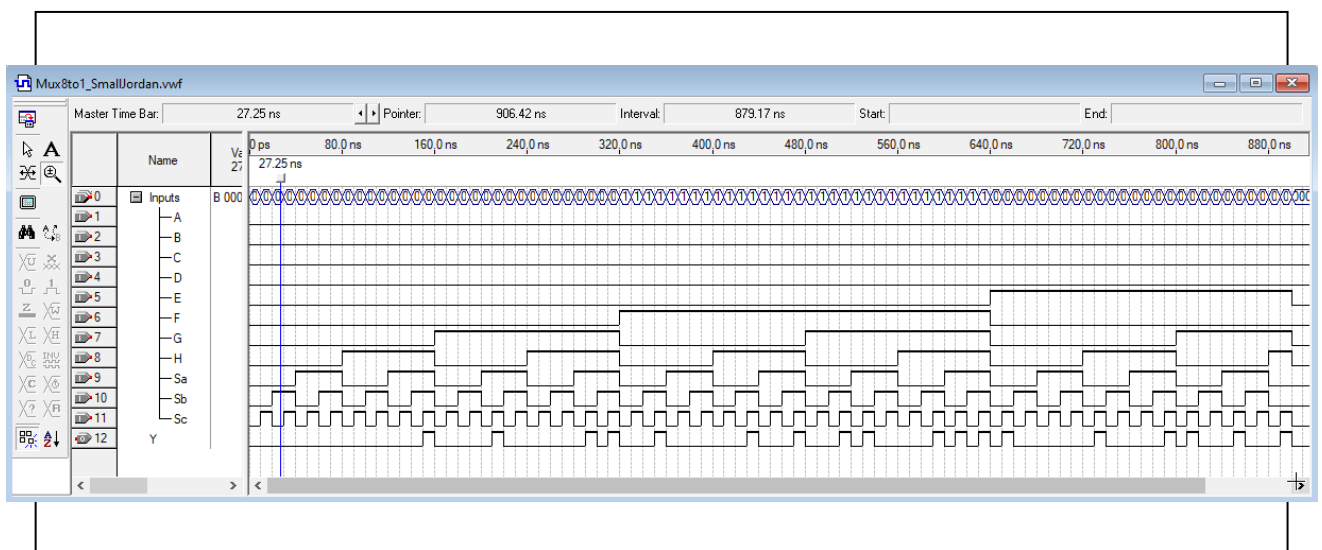
```

Successful Compilation

The screenshot shows the Quartus II IDE with the following components:

- Main Editor:** Displays the Verilog code for `Mux8to1_SmallJordan.vhd`. The code defines a module with inputs `A, B, C, D, E, F, G, H, Sc, Sb, Sa` and output `Y`. It uses a library `ieee` and includes `std_logic_1164.all`. The module is composed of three sub-components: `Mux8to1_SmallJordan`, `Mux2to1_SmallJordan`, and `Mux4to1_SmallJordan`.
- Quartus II Dialog:** A message box indicating "Full Compilation was successful (1 warning)".
- Compiler Tool Window:** Shows the progress of various compilation steps:
 - Analysis & Synthesis:** 100% complete, 00:00:01.
 - Filter:** 100% complete, 00:00:01.
 - Assembler:** 100% complete, 00:00:01.
 - Classic Timing Analyzer:** 100% complete, 00:00:00.
 - Full Compilation:** 100% complete, 00:00:03.

Timing Diagram



2.6 Create a new project in Altera Quartus using VHDL of a **16-to-1 Mux**, call it Mux8to1_YourName, using only Mux8to1, Mux4to1 and/or Mux2to1 components

Project Wizard

New Project Wizard: Summary [page 5 of 5]

When you click Finish, the project will be created with the following settings:

Project directory:
C:/Users/Jordan/Desktop/School/Fall 2023/CDA3203/Lab1_SmallJordan/

Project name: Mux16to1_SmallJordan
Top-level design entity: Mux16to1_SmallJordan
Number of files added: 3
Number of user libraries added: 0

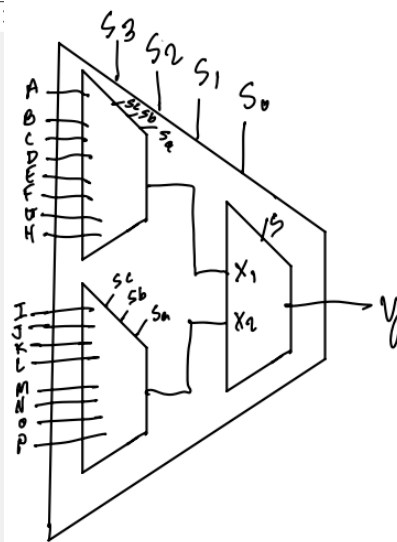
Device assignments:
Family name: MAX3000A
Device: EPM3064ALC44-10

EDA tools:
Design entry/synthesis: <None>
Simulation: <None>
Timing analysis: <None>

Operating conditions:
Core voltage: 3.3V
Junction temperature range: 0-85 °C

< Back Next > Finish Cancel

Mux16to1 Diagram



Truth Table

S ₃	S ₂	S ₁	S ₀	Y
0	0	0	0	A
0	0	0	1	B
0	0	1	0	C
0	0	1	1	D
0	1	0	0	E
0	1	0	1	F
0	1	1	0	G
0	1	1	1	H
1	0	0	0	I
1	0	0	1	J
1	0	1	0	K
1	0	1	1	L
1	1	0	0	M
1	1	0	1	N
1	1	1	0	O
1	1	1	1	P

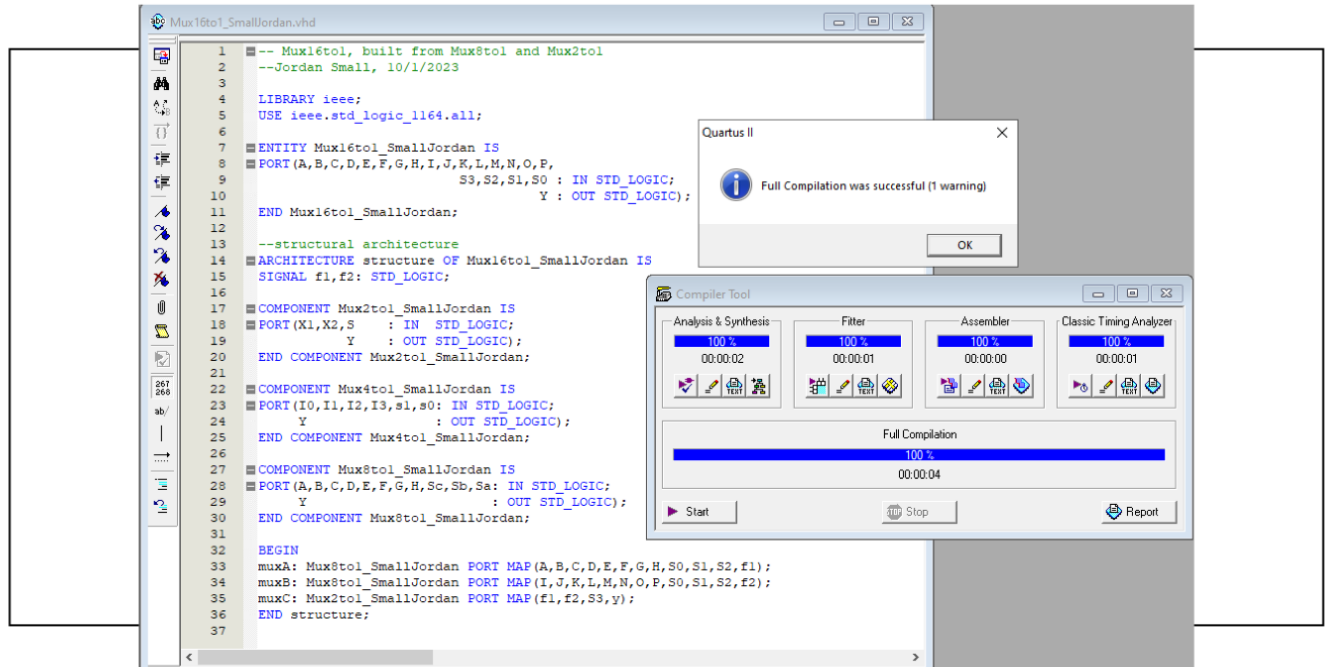
VHDL code

```

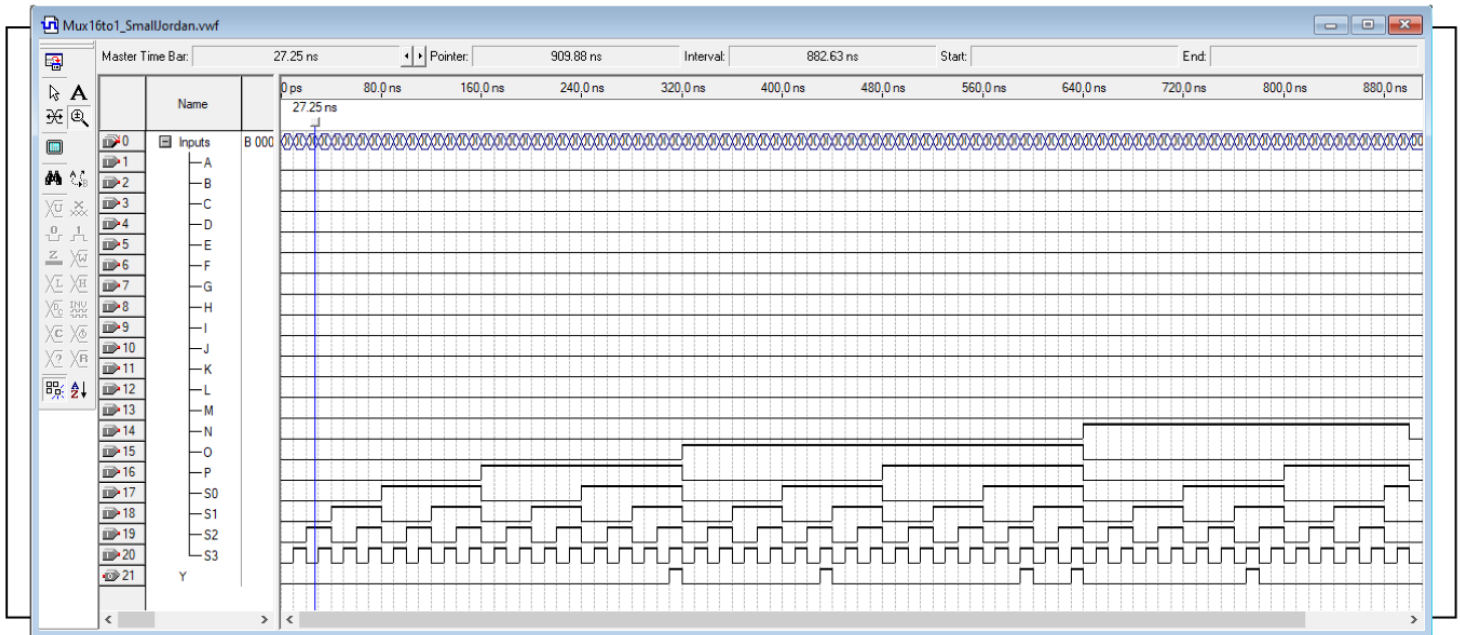
1  -- Mux16to1, built from Mux8to1 and Mux2to1
2  --Jordan Small, 10/1/2023
3
4  LIBRARY ieee;
5  USE ieee.std_logic_1164.all;
6
7  ENTITY Mux16to1_SmallJordan IS
8  PORT (A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,
9        S3,S2,S1,S0 : IN STD_LOGIC;
10        Y : OUT STD_LOGIC);
11  END Mux16to1_SmallJordan;
12
13  --structural architecture
14  ARCHITECTURE structure OF Mux16to1_SmallJordan IS
15  SIGNAL f1,f2: STD_LOGIC;
16
17  COMPONENT Mux2to1_SmallJordan IS
18  PORT (X1,X2,S : IN STD_LOGIC;
19        Y : OUT STD_LOGIC);
20  END COMPONENT Mux2to1_SmallJordan;
21
22  COMPONENT Mux4to1_SmallJordan IS
23  PORT (I0,I1,I2,I3,s1,s0: IN STD_LOGIC;
24        Y : OUT STD_LOGIC);
25  END COMPONENT Mux4to1_SmallJordan;
26
27  COMPONENT Mux8to1_SmallJordan IS
28  PORT (A,B,C,D,E,F,G,H,Sc,Sb,Sa: IN STD_LOGIC;
29        Y : OUT STD_LOGIC);
30  END COMPONENT Mux8to1_SmallJordan;
31
32  BEGIN
33  muxA: Mux8to1_SmallJordan PORT MAP (A,B,C,D,E,F,G,H,S0,S1,S2,f1);
34  muxB: Mux8to1_SmallJordan PORT MAP (I,J,K,L,M,N,O,P,S0,S1,S2,f2);
35  muxC: Mux2to1_SmallJordan PORT MAP (f1,f2,S3,y);
36  END structure;

```


Successful Compilation



Timing Diagram



This project was done by: Jordan Small

following the tutorial videos created by Dr. Petrie and material in the class textbook, with no other outside resources or help with the following exceptions:

x Received help from Teaching Assistant: Harry (Timing diagrams for Mux8to1 and Mux16to1; he said no annotation was necessary for these)

___ Found the following material on the internet: show URL

___ Other: